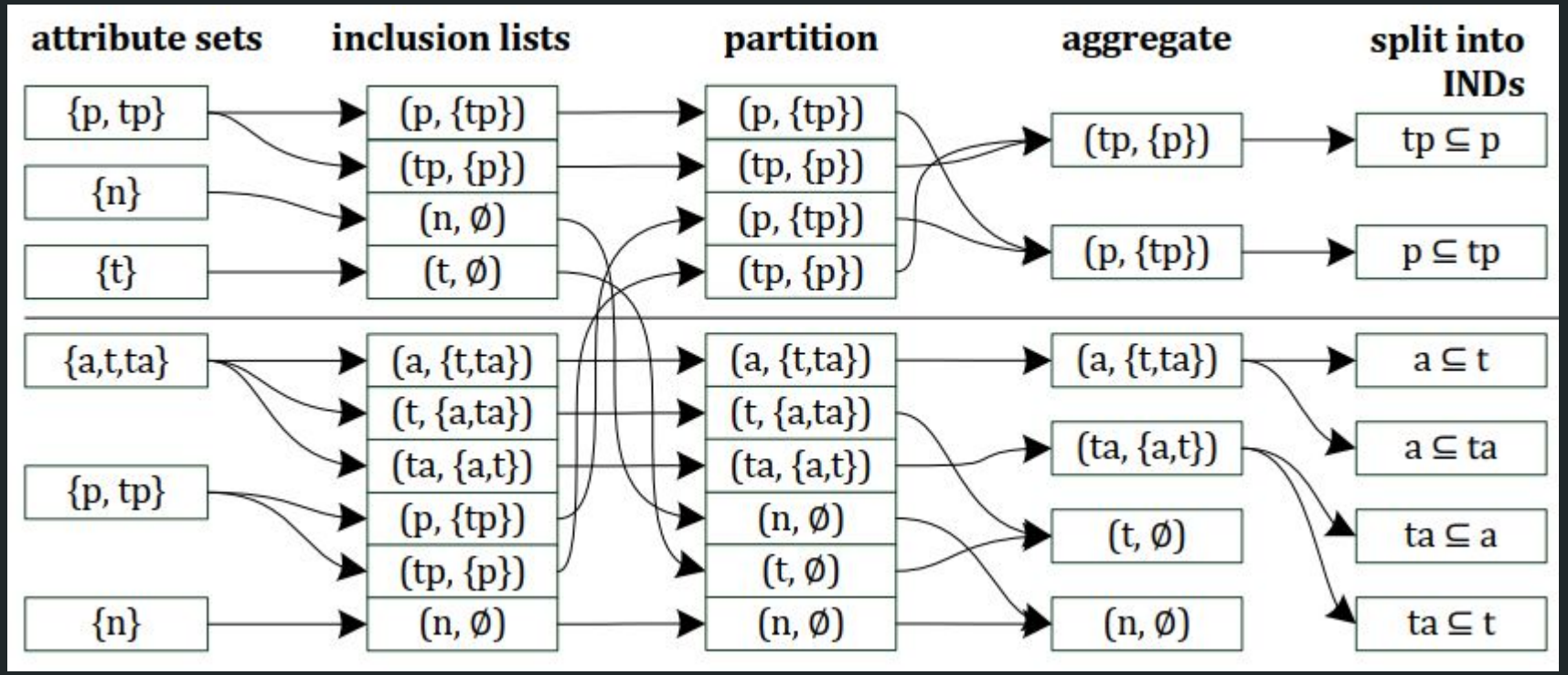
Spark-homework

Team “nescimus”

Architecture



From: Kruse, Sebastian, Thorsten Papenbrock, and Felix Naumann. "Scaling out the discovery of inclusion dependencies." Datenbanksysteme für Business, Technologie und Web (BTW 2015) (2015).

Code

- input read to format `List[Dataset[(String, Set[String])]]`

```
inputs
  .map(input => readData(input, spark))
  .map(ds => ds.flatMap(row => row.schema.names
    .map(name => (row.getString(row.fieldIndex(name)), Set(name)))))
```

- union tuples, group by Key, put corresponding colNames into Sets

```
  .reduce(_.union(_))
  .groupByKey(x => x._1)
  .mapValues(x => x._2)
  .reduceGroups(_.union(_))
  .filter(set => set._2.size > 1)
```

Code

- create possible INDs

```
var mapping = datasink.map(x => x._2)
  .flatMap(x => x.map(v => (v, x-v)))
var aggregate = mapping
  .groupByKey(x => x._1)
  .mapValues(x => x._2)
```

- reduce to set of possible INDs per Column, filter relevant ones

```
  .reduceGroups((k, v) => k.intersect(v))
  .filter(x => x._2.size > 1)
```

- print List of cleaned Strings for output

```
var b = ind
  .map(entry => (entry._1 + " < " + entry._2.toString()
    .slice(4, entry._2.toString().length-1)))
  .collect().sorted
b.foreach(println(_))
```